

Extracting Keywords with Combination of Text Clustering by Graph based KNN and AHC Algorithm

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the graph based KNN algorithm and the graph based AHC algorithm for automating the keyword extraction by introducing the text clustering. In the previous work, even if the graph based KNN algorithm was applied for the keyword extraction, it is required to present a domain as an input text tag for carrying out the task. In this research, a text group is made into text subgroups based on their contents by clustering texts, and each word in a novice text is classified into keyword or non-keyword, by a classifier which corresponds to its most similar cluster. The AHC algorithm and the KNN algorithm which process graphs directly are applied respectively as a text clustering tool and a keyword extraction tool. The goals of this research are to automate the keyword extraction and to improve the performance of both tasks, using the graph-based versions of the two algorithms.

I. INTRODUCTION

The keyword extraction is defined as the process of extracting automatically essentially important words from a text. It is interpreted into a binary classification where the two categories, keyword and non-keyword, are predefined. The process of keyword extraction is to index a text into words, classify each word into one of the two categories, and extract ones which are classified into keyword. The classification task which is mapped from the task is a domain dependent task where a same word may be classified differently depending on the domain. It is necessary to present the domain together with the input text for performing the keyword extraction.

This research is motivated by the necessity of defining domains in advance for designing the keyword extraction system. The keyword extraction is interpreted into domain dependent classifications; the keyword extraction system is composed with binary classification systems as many as domains. The process of designing the keyword extraction system is to decide domains as a list, collect sample examples domain by domain, and allocate a classifier for each domain. The results from applying the graph based KNN algorithm and the graph based AHC algorithm respectively to the keyword extraction and the text clustering were successful. This research is intended to automate the process of defining domains by connecting the keyword extraction with the text clustering.

The idea of this research is to connect the keyword extraction with the text clustering and apply the graph based KNN algorithm and the graph based AHC algorithm to both tasks. In this research, it is assumed that the collection initially consists of texts, each of which is associated with its own keywords. The similarity metric between graphs is defined, and the KNN algorithm and the AHC algorithm are modified into the graph-based version. The collection is partitioned into clusters by the text clustering, each cluster corresponds to its own domain, and the keyword extraction system is implemented cluster by cluster. A domain is automatically decided to an input text by its similarities with the cluster prototypes for selecting a corresponding keyword extraction system.

Let us mention some benefits from this research. The main problems in encoding words and texts into numerical vectors are solved by encoding them into graphs. The performance is improved by adopting the graph-based versions of the KNN algorithm and the AHC algorithm as the approaches to the keyword extraction and the text clustering. The domains are predefined through the text clustering automatically; prior knowledge or subjective views are not needed for defining the domains. A domain is automatically decided by the similarities of an input text with the cluster prototypes; only an input text is presented without its domain for nominating a binary classifier.

Let us mention some remaining tasks as further discussions on this research. More advanced machine learning algorithms and deep learning algorithms should be modified into the graph-based versions. It is necessary to define and characterize mathematically other operations on graphs for doing that. The keyword extraction system which relates to the text clustering should be implemented as a real program or a library. The keyword extraction may relate to other text mining tasks for improving their performances.

II. PROPOSED APPROACH

A. Similarity Metric

This section is concerned with the graph as a word representation and the similarity between graphs. In representing a word into a graph, a vertex indicates a text, and an

edge indicates a similarity between texts. The similarity between edges is computed, before computing the similarity between graphs, viewing a graph as a set of edges. The similarity between graphs is computed by averaging over the similarities of all possible pairs of edges. In this section, we describe the process of encoding a word into a graph and the process of computing the similarity between graphs.

The process of encoding words into graphs is illustrated in Figure 1. In the vertex definition, from a text collection, texts which are relevant to the word are retrieved as the vertices. In the edge definition, the similarities among the retrieved texts are computed as edges. In the edge selection, the edges with their higher similarities are selected for representing a graph. Refer to [1] for studying the encoding process in more detail.

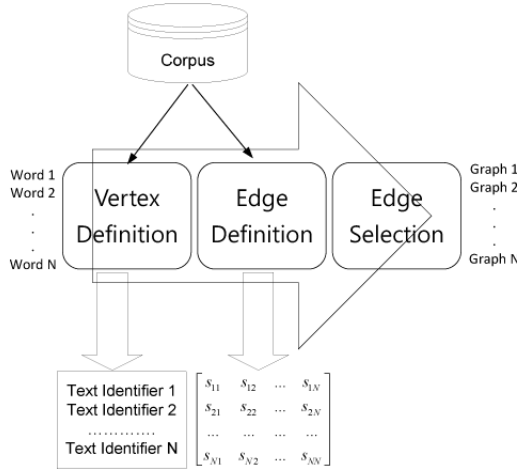


Figure 1. Process of Encoding Words into Graphs

The three cases of computing the similarity between two edges are illustrated in Figure 2. Each weight which is associated with its own edge is assumed to be a normalized value between zero and one, and if both nodes are identical to each other, the average over two weights is the similarity between two edges. If either of the two nodes is identical to each other, the product of two weights is the similarity between two edges. If neither of two nodes is identical, zero is the similarity between them. The similarity between two edges is always a normalized value between zero and one.

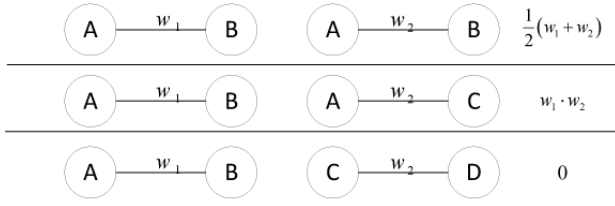


Figure 2. Three Cases of Comparing Two Edges with Each Other

Let us mention the process of computing the simi-

ilarity between graphs as a semantic similarity between words. The two graphs which represent words are notated by edge sets, $G_1 = \{e_{11}, e_{12}, \dots, e_{1|G_1|}\}$ and $G_2 = \{e_{21}, e_{22}, \dots, e_{2|G_2|}\}$. Two edges, $e_{1i} \in G_1$ and $e_{2j} \in G_2$, are associated with their own weights, w_{1i} and w_{2j} , and the similarity between two edges, e_{1i} and e_{2j} by equation (1), (2), or (3), depending on the cases which are mentioned above,

$$\text{sim}(e_{1i}, e_{2j}) = \frac{1}{2}(w_{1i} + w_{2j}) \quad (1)$$

$$\text{sim}(e_{1i}, e_{2j}) = w_{1i} \times w_{2j} \quad (2)$$

$$\text{sim}(e_{1i}, e_{2j}) = 0 \quad (3)$$

The similarity between two graphs is computed by equation (4),

$$\text{sim}(G_1, G_2) = \frac{1}{|G_1| \times |G_2|} \sum_{i=1}^{|G_1|} \sum_{j=1}^{|G_2|} \text{sim}(e_{1i}, e_{2j}). \quad (4)$$

The value which is generated from equation (4), is always given as a normalized value between zero and one as shown in equation (5),

$$0 \leq \text{sim}(G_1, G_2) \leq 1. \quad (5)$$

Let us make some remarks on the process of encoding a word into a graph and computing the similarity between graphs. A word is encoded into a graph by the vertex definition, the edge definition, and the edge weighting. The three cases are considered for computing the similarity between edges. The similarity between graphs is computed by equation (4). In the future research, we consider representing a word into multiple graphs.

B. Graph based KNN Algorithm

This section is concerned with the graph based KNN algorithm. It was initially proposed as the approach to the word classification by Jo in 2018 [1]. The similarity metric is used for computing the similarity between a novice graph and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into graphs, and they are notated by a set, $Tr = \{G_1, G_2, \dots, G_N\}$. A novice word is encoded into a graph notated by G . Its similarities with the graphs which represent the sample words are computed by equation (4), as $\text{sim}(G, G_1), \text{sim}(G, G_2), \dots, \text{sim}(G, G_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

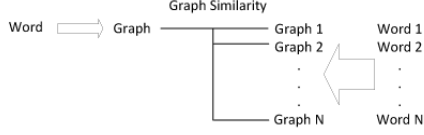


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (6),

$$Nr = \text{Select}_k(Tr, G), Nr \subseteq Tr \quad (6)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (7),

$$Nr = \{G_1^{near}, G_2^{near}, \dots, G_k^{near}\} \quad (7)$$

The elements in the set, Nr , are used for voting their labels in the next step.

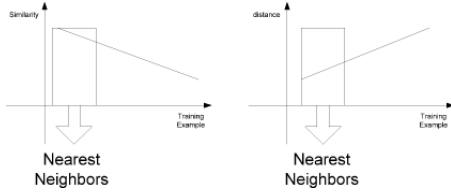


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(G_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, G , is decided by equation (8),

$$\text{label}(G) = \underset{j=1}{\text{argmax}}^m \text{Count}(Nr, c_j) \quad (8)$$

The process of deciding the label of a novice item by equation (8), is called voting.

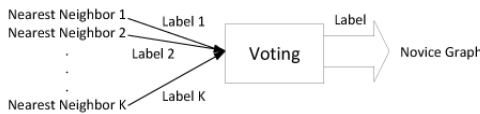


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the graph-based version of KNN algorithm. It computes the similarities of a novice item

with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the keyword extraction system which adopts the graph based KNN algorithm. In Section II-B, we described the proposed version of KNN algorithm as the approach to the keyword extraction. It is viewed as a binary classification which classifies a word into keyword or non-keyword. This section is intended to describe the keyword extraction system with respect to its function and its architecture.

The process of collecting the sample words and encoding them into graphs for implementing the keyword extraction system is illustrated in Figure 6. The keyword extraction is viewed as a binary classification which classifies a word into keyword or non-keyword. The topic-based word classification belongs to the domain independent classification where a same word is always classified identically with regardless of the domain, whereas the keyword extraction is the domain dependent classification where a same word may be classified differently depending on the domain. The words which are labeled with keyword or non-keyword are collected domain by domain. It is required to tag the input text with its own domain for executing the keyword extraction.

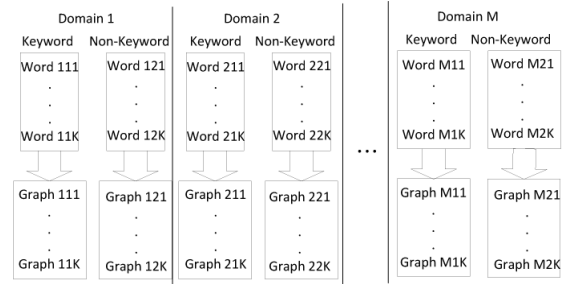


Figure 6. Preparation of Training Examples

The entire architecture of the proposed keyword extraction system is illustrated in Figure 7. A text is given as the input, and words are extracted from it in the indexing module. In the encoding module, sample words in the keyword group and the non-keyword group and ones which are indexed from the text are mapped into graphs. The words which are indexed from the text are classified into one of the two categories in the similarity computation module and the voting module. The system consists of the four modules: indexing module, encoding module, similarity computation module, and voting module.

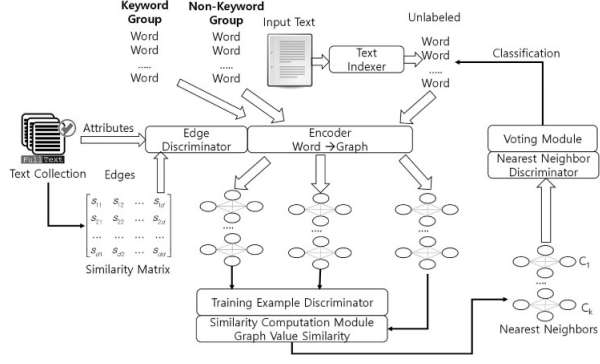


Figure 7. System Architecture

The execution process of the proposed system is illustrated as a block diagram in Figure 8. Sample words which are labeled with keyword or non-keyword are collected within a domain, and they are encoded into graphs. An input text is indexed into a list of words, and they are also encoded into graphs. The nearest neighbors are selected by the similarity computation, and the label is decided by voting ones of their nearest neighbors for each word. The words which are classified into keyword are extracted as the final output.

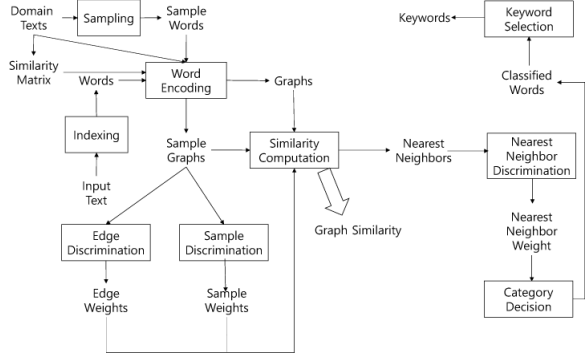


Figure 8. System Flow

Let us make some remarks on the proposed system whose architecture is presented in Figure 7. The keyword extraction is defined as a binary classification where each word is classified into keyword or non-keyword. Each word is encoded into a graph instead of a numerical vector and is classified directly. Among words which are included in the input text, ones which are classified into keyword are selected. We need to add the text classification module for predicting the domain of the input text automatically.

D. Connection with Text Clustering

This section is concerned with the connection of the keyword extraction with the text clustering. In the previous section, we mentioned the keyword extraction as an instance of domain dependent classification. The domains

are automatically constructed from the text collection by clustering texts. The AHC algorithm which is proposed in [3] is used for implementing the text clustering. This section is intended to describe the keyword extraction system which is connected with the text clustering for automating the construction of domains.

The architecture of the text clustering system which was proposed in [3] is illustrated in Figure 9. The texts in the group are encoded into graphs, and the AHC algorithm which is proposed in [3] is adopted as the approach. It starts with singletons as many as items, computes the similarities of all possible cluster pairs, and merge the cluster pair with its highest similarity into one cluster. The two steps are iterated until the number of clusters reach the desired number. We may consider replacing the AHC algorithm by its variants for improving the speed and the performance.

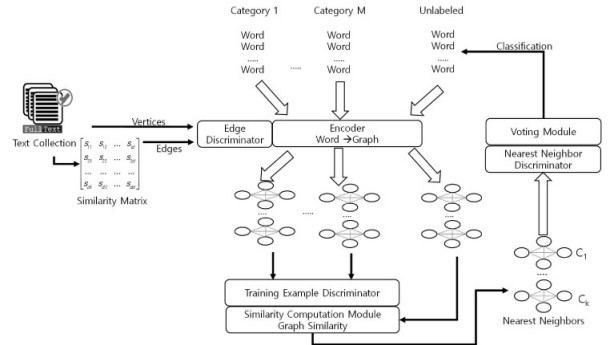


Figure 9. Text Clustering System

The connection of the keyword extraction with the text clustering is illustrated in Figure 10. The M domains are defined by clustering texts in a group, initially and automatically. A novice text is given as the input, and its domain, domain D , is decided by its similarities with domains. A classifier which corresponds to domain D is nominated and classify each word in a novice text into keyword or non-keyword. The M keyword extraction systems are viewed as independent ones, each of which classifies each word into one of the two categories.

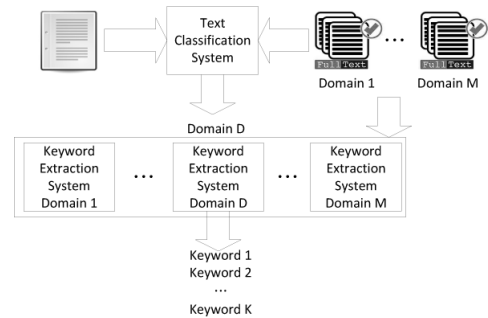


Figure 10. Keyword Extraction System connected with Text Clustering

The execution flow of keyword extraction which is connected with the text clustering is illustrated in Figure 11. Unlabeled texts are clustered into subgroups, each of which is a domain. Sample words which are labeled with keyword or non-keyword are generated from each domain. These sample words are provided for training the classifier in each keyword extraction system. Each classifier is used for judging whether each word which is indexed from the input text is keyword or not.

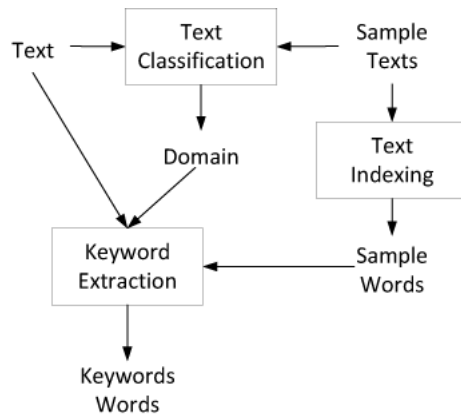


Figure 11. System Flow of Keyword Extraction connected with Text Clustering

Let us make some remarks on the connection of the keyword extraction with the text clustering. It is the process of segmenting a text group into subgroups based on their content-based similarities. The role of text clustering is to define the domains initially in the architecture of connecting the keyword extraction with the text clustering. In each domain, sample words are generated, and its corresponding classifier is trained with them. In the future research, we consider using the keyword extraction for clustering texts in the opposite direction.

REFERENCES

- [1] T. Jo, "Comparing Graph based K Nearest Neighbor with Traditional Version in Word Categorization in NewsPage.com, 12-18, International Journal of Advanced Social Sciences, 1, 2018.
- [2] T. Jo, "Graph based K Nearest Neighbors for Keyword Extraction in Current Affair Domain", 47-48, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.
- [3] T. Jo, "Graph based Version for Clustering Texts in Current Affair Domain", 171-176, The Proceedings of 15th International Conference on Data Science, 2019.